université
PARIS-SACLAY

PIIA Licence 3

Projet Groupe 2 :
Agenda Étudiant Jardinage
RAPPORT FINAL

Sodiki Olawale Salami
Albanio De Souza

Tables des matières :

A. Introduction

Après avoir bien préparé les fonctionnalités nécessaires à l'application ainsi que les différents cas d'utilisation, nous allons programmer notre agenda. Pour cela, nous suivons le plus possible le cahier des charges du rapport précédent. Etant donné que nous ignorions quelles bibliothèques nous allions utiliser, il y aura quelques modifications que nous préciserons plus tard.
En ce qui concerne le choix des bibliothèques, nous avons choisi JavaFX pour sa qualité graphique, de plus, notre chargé de TD nous l'a recommandé et nous l'avons utilisé pendant les TP. Pour la gestion du planning, il nous a conseillé CalendarFX ainsi que des fichiers "Json" pour la sauvegarde des informations de l'utilisateur.

B. Fonctionnalités

a. Fonctionnalités prévues

Fonctionnalités	Définition	Mode d'utilisation prévu / Etat actuel
Ajout et édition d'une nouvelle plante	Il s'agit d'une fonctionnalité permettant aux étudiants utilisateurs de pouvoir ajouter autant de plantes que d'activités de jardiniers.	Il suffira ici de cliquer sur le bouton d'ajout de nouvelle plante. Cela permettra d'accéder à l'interface dédiée. Sur cette interface il y aura des formulaires permettant d'effectuer l'opération d'ajout (Photo, informations..) et d'édition. **Réalisé sauf l'ajout d'image(*1)**

Créer des évènements (Ponctuels, récurrents)	Une fonctionnalité importante dans le suivi des plantes. Elle permettra d'ajouter des évènements tels que la récolte, Nettoyage, …), plus tard de permettre de mettre en place un système de planification et par suite des rappels. Fonctionnalité possible également à travers la page de la plante.	Un bouton dédié sera sur l'interface applicative, également accessible sur la page de la plante, il suffira d'effectuer un clic afin d'accéder au formulaire de création d'évènement. **Réalisé différemment (*2)**
Liste de plante	Elle permettra aux étudiants de visualiser la totalité des plantes qu'ils gèrent.	Clic simple sur le bouton "Liste des plantes" Chaque image de plante sera cliquable afin d'afficher une page dédiée à la plante en question. **Réalisé**
Ajout de besoin	Cette fonctionnalité servira à définir les éventuels besoins des plantes (En eau, …). Accessible à travers la page de la plante ou à travers l'interface.	Un étudiant peut ajouter les besoins d'une plante en cliquant sur le bouton "besoin" . **Non réalisé sur demande du prof (fonctionnalité non nécessaire pour ce type de cibles)**
Gestion de suivie	Permet de faciliter le suivi des différentes étapes d'évolution de la plante.	Accessible à travers la page de la plante, il suffira d'effectuer un clic sur le bouton d'ajout de de suivi pour ajouter les informations de suivi. **Réalisé**
Planning	Permet la visualisation des différents évènements planifiés.	Il s'agit de la première page afficher sur l'interface. Le clic sur le bouton planning va permettre de revenir sur la page. **Réalisé**
Liste de suivi	Permet de visualiser les différentes étapes de suivi de la plante, aussi d'accéder à la page des graphes de mesure de la plante.	Clic sur le bouton liste suivi, accessible via la page de la plante. **Réalisé**
Liste mesure	Fonction permettant la visualisation de l'avancée des proportions de la plante à travers des graphes.	Un simple clic sur le bouton liste mesure. **Réalisé mais sans le graphe**
Création de projet	Permet à l'étudiant d'ajouter un nouveau projet. Accessible depuis la liste des projets.	Un clic sur le bouton "+". **Retiré par manque de temps**

Liste des projets	Fonction permettant d'afficher tous les projets (en cours en haut de liste et terminés en bas de liste)	Un clic sur le bouton "liste des projets", accessible via la page de la plante. **Retiré par manque de temps**
Affectation de projet	Cette fonctionnalité servira à définir les éventuels projets des plantes. Accessible à travers la page de la plante.	Un étudiant peut affecter un projet à une plante en cliquant sur le bouton "+ projet". **Retiré par manque de temps**
Supprimer une plante d'un projet	Cette fonctionnalité servira à retirer une plante d'un projet. Accessible à travers la page de la plante ou de la page d'un projet.	Dans la page de la plante une croix permet (après demande de confirmation de supprimer la plante du projet) et de même lorsqu'on est dans la page du projet on a une croix à côté de la plante. **Retiré par manque de temps**

(*1) : Gestion d'images aléatoires non choisies par l'utilisateur.
(*2) : Ajout d'événements via un bouton et non depuis la page de la plante comme prévu.

b. Fonctionnalités ajoutées

Fonctionnalités	Définition	Mode d'utilisation
Recherche de plante	Rechercher un plante à l'aide de son nom	Dans l'onglet liste des plantes, on peut saisir le nom exacte de la plante que l'on recherche puis cliquer sur le bouton rechercher
Ajout de zone	Possibilité d'ajout de nouvelles zones	Bouton nouvelle zone permettant de créer des zones
Liste des évènements	Consulter tous les évènements existants	Bouton liste des évènements permettant de consulter ceux qui existent (possibilité de trier par ordre croissant/ décroissant selon le nom, la date etc.)
Afficher un évènement	Chaque événement créé se voit afficher dans le calendrier.	L'Affiche se fait automatiquement.
Créer une activité	Possibilité de créer une activité	Bouton créer une activité en ajoutant son nom puis en l'enregistrant.
Créer un suivi	Possibilité de créer un suivi d'une plante	Depuis l'onglet de la plante, on peut créer un nouveau suivi pour celle-ci

C. Code source et application

Extraits de script du projet

Remplir Grid
Cette fonction permet de remplir la grille de la liste des plantes en fonction du nom de la plante saisie dans le formulaire de recherche de la plante.

```
public void remplirGrid(String nom) throws IOException, ParseException {
    Plantes pl = new Plantes();
    String Id = pl.getId(nom);
    /static-access/
    JSONObject planteJsOb = pl.getData(Id);
    System.out.println(planteJsOb);
    pl.setNom(planteJsOb.get("Nom").toString());
    pl.setVariete(planteJsOb.get("Variete").toString());
    pl.setPoids(Double.parseDouble(((JSONObject)planteJsOb.get("Mesures")).get("Poids").toString()));
    pl.setUrl(planteJsOb.get("Url").toString());
    listPlante.clear();
    GradPane.getChildren().clear();
    listPlante.add(pl);

    FXMLLoader fxl = new FXMLLoader();
    fxl.setLocation(getClass().getResource( name: "/views/ItemPlante.fxml"));
    AnchorPane anchorpane = fxl.load();
    ItemPlanteControl itemPlanteCtr = fxl.getController();
    itemPlanteCtr.setData(listPlante.get(0));

    GradPane.add(anchorpane,  i: 0,  i1: 0);
    GridPane.setMargin(anchorpane, new Insets( v: 10));
}
```

Enregistrer une nouvelle zone (Enregistrer Opération)
Fonction d'enregistrement d'une nouvelle zone.

```
@FXML
void Annuler() {
    txtNom.setText("");
    txtTaille.setText("");
    type.setText("");
    cmbType.setValue("");
}

/unchecked/
@FXML
void EnregistreOperation() throws IOException, ParseException {
        Zones zone = new Zones();
        zone.setId(zone.getLastId()+1);
        zone.setNom(txtNom.getText());
        zone.setTaille(Double.parseDouble(txtTaille.getText()));
        zone.setType((String)cmbType.getValue());

        JSONArray listezones = ReadWriteFileJson.readerFileJson( fileName: "Zones");
        listezones.add(zone.tranformZoneToObjectJson());
        ReadWriteFileJson.writeFileJson( fileName: "Zones", listezones);
        ReadWriteFileJson.showAlertWithHeaderText( title: "Ajout Zone",  msg: "La zone a bien été crée");
        Annuler();
}
```

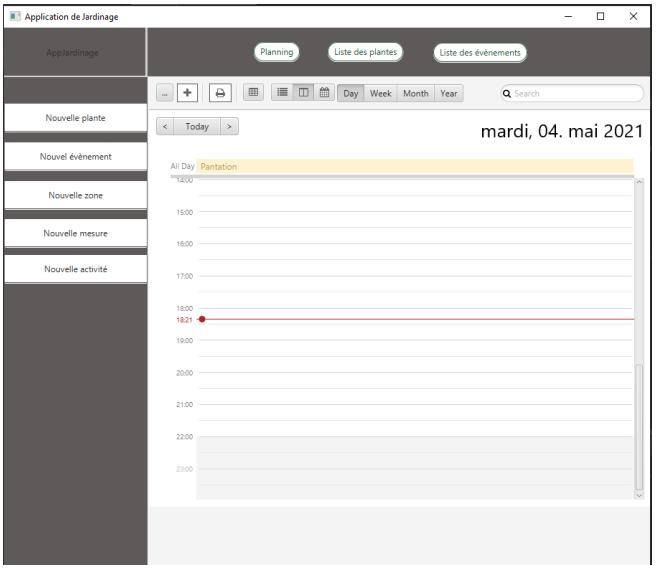
Enregistrer une plante

Fonction d'enregistrement d'une nouvelle plante.

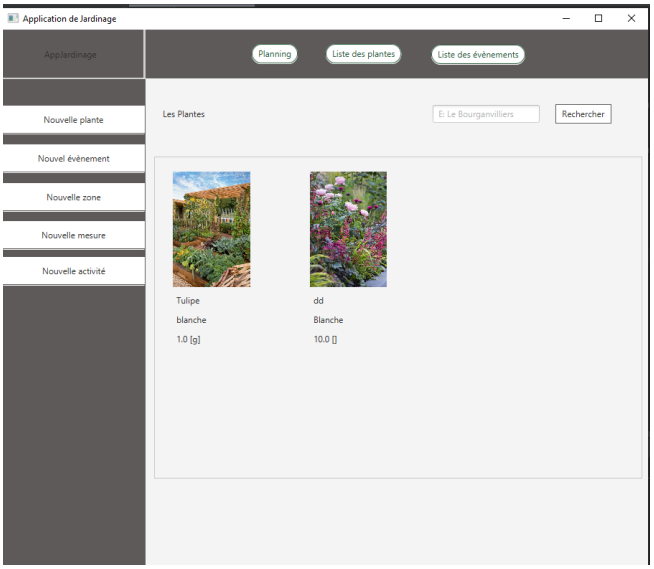
```java
@FXML
void Enregistrer() throws IOException, ParseException {
    Events event = new Events();
    event.setId(event.getLastId()+1);
    event.setActivite(listeActivite.getValue());
    event.setDate(datePicker.getValue().toString());
    event.setNote(txtAreaNote.getText());
    event.setTypeEvent(listeTypeEvent.getValue());


    if(listePlante.getValue()!=null) {
        JSONArray eventsPlantes = ReadWriteFileJson.readerFileJson( fileName: "EventsPlantes");
        JSONObject eventP = new JSONObject();
        eventP.put("IdEvent", event.getId());
        Plantes pl= new Plantes();
        eventP.put("IdPlantes", pl.getId(listePlante.getValue()));

        eventsPlantes.add(eventP);
        ReadWriteFileJson.writeFileJson( fileName: "EventsPlantes", eventsPlantes);
    }

    JSONArray liste_events = ReadWriteFileJson.readerFileJson( fileName: "Events");
    liste_events.add(event.tranformEventToObjectJson());
    ReadWriteFileJson.writeFileJson( fileName: "Events", liste_events);
    ReadWriteFileJson.showAlertWithHeaderText( title: "Création de l'évènement",  msg: "Votre évènement a bien été enregistrer");


    try {
        list.getData();
    } catch (IOException | ParseException e) {
        e.printStackTrace();
    }
}
```

Extraits de l'application

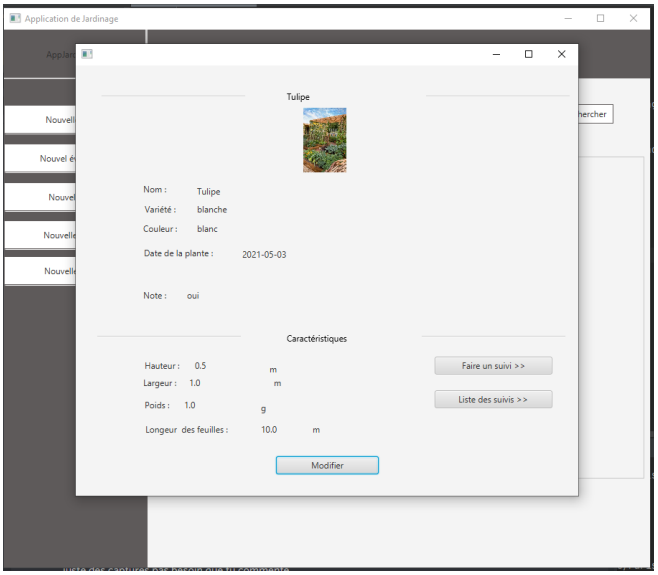
Page planning :



Page liste des plantes :



Page d'une plante :



D. Installation

(JavaFX supposé installé)
Dézipper le fichier après l'avoir télécharger
Ajouter les fichiers ***.Jar** situés dans dossier src/jars
Ajouter également les fichiers ***.Jar** situés dans dossier src/jars/calendarFx
Lancer l'application.

E. Conclusion

Nous avons créé un agenda permettant à un étudiant jardinier de gérer ses plantes, d'y ajouter des mesures ou des suivis, d'avoir un planning ou visualiser des évènements qu'il crée lui-même. L'application est simple d'utilisation afin qu'une personne ne maîtrisant pas l'informatique puisse l'utiliser facilement. Les fonctionnalités les plus importantes ont été créées mais certaines n'ont pas été ajoutées par manque de temps. Notamment l'ajout de projet, de météo ou encore d'affichage d'images choisies par l'utilisateur. L'interface pourrait être moins basique également. Mais l'application prévoit de futures mises à jour et sera bientôt disponible gratuitement sur Android et Apple.